TDB-ACC-NO: NN77122921

DISCLOSURE TITLE: Technique to Fill a Complex Polygon With Rectangles.
December 1977.

PUBLICATION-DATA: IBM Technical Disclosure Bulletin, December 1977, US

VOLUME NUMBER: 20

ISSUE NUMBER: 7

PAGE NUMBER: 2921 - 2924

PUBLICATION-DATE: December 1, 1977 (19771201)

CROSS REFERENCE: 0018-8689-20-7-2921

DISCLOSURE TEXT:

4p. A common problem in computer graphics and mask design is to fill a complex polygon, which may contain numerous concave corners or holes, with simpler figures, usually rectangles. The technique herein described is designed to fill a complex polygon consisting of horizontal and vertical lines with the minimum number of rectangles.

- An economical way of describing a complex polygon is by subtracting small rectangles from a large rectangle. For example, as shown in Fig. 1, the desired resultant polygon is rectangle A minus rectangles B, C and D.

- All the rectangles can be described by the coordinates of their lower left corner and upper right corner, respectively.

- A technique for filling a complex polygon with rectangles is as follows:

1. Read in a list of rectangle descriptors, consisting of the

lower left and upper right corner coordinates. A typical set, where the large background rectangle is given first, is:

TABLE

RECT. #	XLL	YLL	XUR	YUR
1	1.	1.	8.	8.
2	2.	2.	4.	5.
3	4.	3.	6.	6.
4	1.	6.	6.	8.

- 2. Form tables of unique values of X-coordinates and unique values of Y-coordinates from the input data. Sort this table in ascending values. Typically for the input data of the above table:

Sorted X-coordinates: 1., 2., 4., 6., 8.

Sorted Y-coordinates: 1., 2., 3., 5., 6., 8.

- 3. Construct a grid wherein the grid coordinates have the values of the sorted X and Y values, as determined in step 2. A typical grid is shown in Fig. 2 for the data in the table.

- Henceforth, each grid box can be addressed by the horizontal and vertical count of box positions from the origin. A typical grid box is shown in Fig. 3, where I and J are the box counts for the horizontal and vertical directions, respectively. Assign to each grid box a variable G(I,J), which is used to indicate whether it is filled or empty. When G(I,J)=1, the box is filled; when G(I,J)=0, the box is empty. Initialize all elements of the array G to 1.

- 4. Skipping the first input rectangle, test the midpoint of each grid box to determine if it falls inside any of the input rectangles. Set the corresponding G(I,J) to 0 if the midpoint falls inside any rectangle except the first. A typical result corresponding to the data in the table is shown in Fig. 4.

- 5. At this point, the complex polygon has been

filled with rectangles which are those grid boxes with a G(I,J) of the value 1. The number of rectangles is not yet minimized.

- 6. The maximum value of I is NI, and the maximum value of J is NJ. For the situation in Fig. 4, NI=4 and NJ=5. Take each grid box one at a time, indexing over I, from 1 to NI, and indexing over J, from 1 to NJ. Scan the grid in the positive I direction and the positive J direction from the current (I,J) position, and determine how many continuous filled grid boxes extend in each direction. IRX is the number of continuous grid boxes in the I, or X, direction, and JRY is the number in the J, or Y, direction.

- In Fig. 4, for grid box (1,1), IRX=4 and IRY=4. Take as a "fill rectangle" the continuous string of grid boxes which have the greatest length originating at the current grid box. If IRX > JRY, the string is horizontal. If JRY > IRX, the string is vertical. Store the lower left and upper right coordinates of this rectangle. Set the value G(I,J) of any grid box included in this rectangle to zero. Go on to the next grid box with G(I,J)=1, and repeat this step until all grid boxes have been processed or the entire array G is set to zero. Fig. 5 shows the fill rectangles derived from Fig. 4. Fig. 6 is the dimensionally correct representation of the fill rectangles.

- 7. Test all fill rectangles to determine if any fill rectangle abuts another which has the same dimension along the abutment. If this happens, merge them into a single fill rectangle and test this new rectangle for abutment with any other fill rectangle. Typical abutments are shown in Fig. 7.

- The procedure described above will find the minimum number of fill rectangles for all configurations of complex polygons anticipated. The minimum number will not be found when multiple singularities occur, such as in Fig. 8. However, to improve upon this would require a higher level of testing which may not be worth the expected improvement.

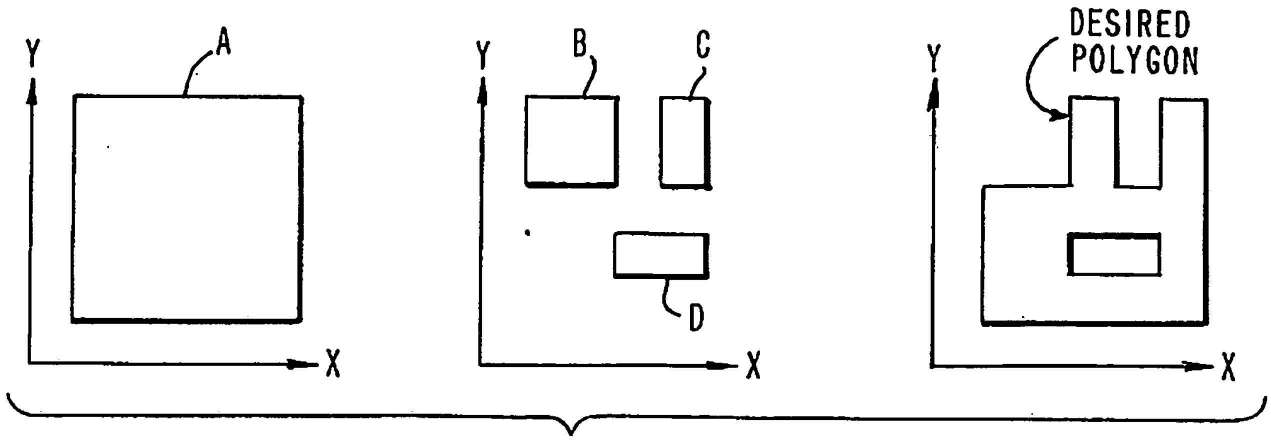
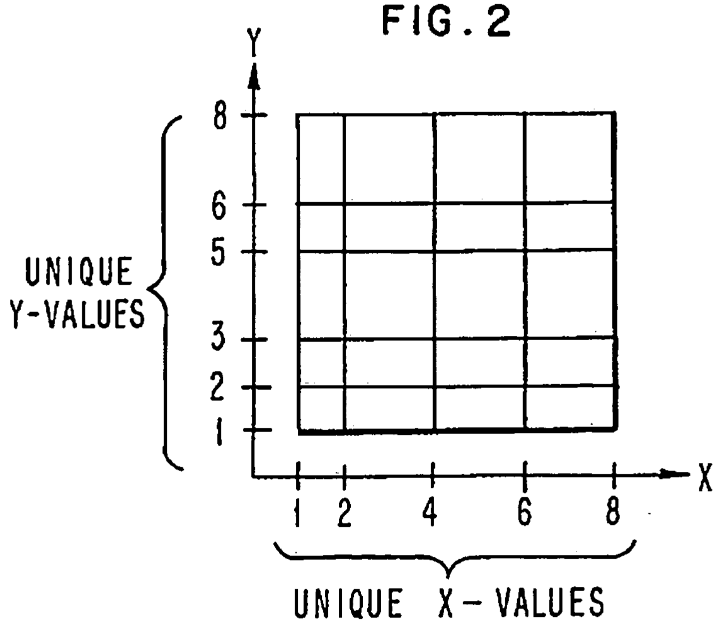


FIG. 1

FIG. 2



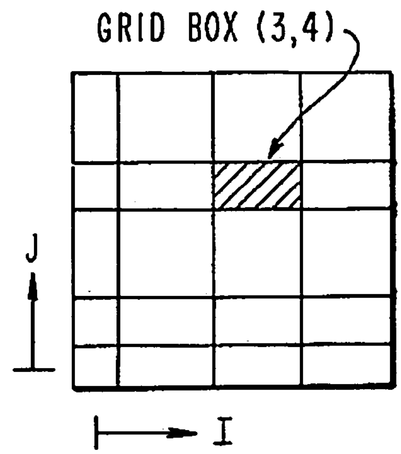


FIG. 3

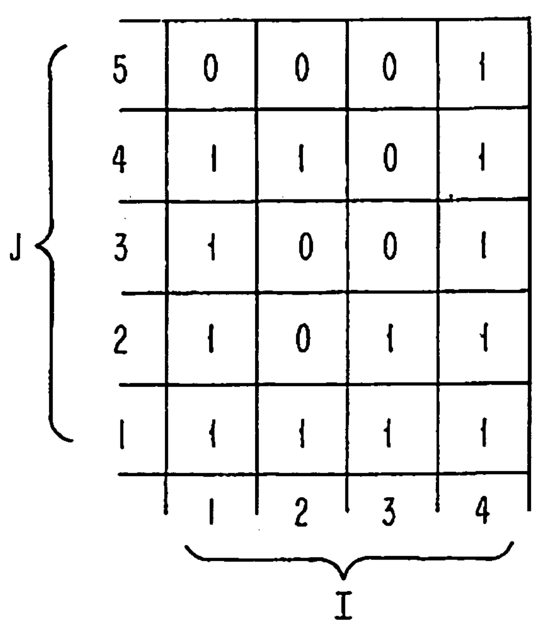


FIG. 4

FIG. 5

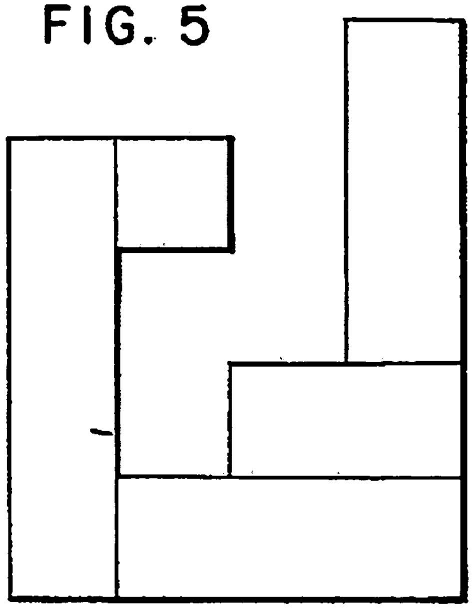


FIG. 6

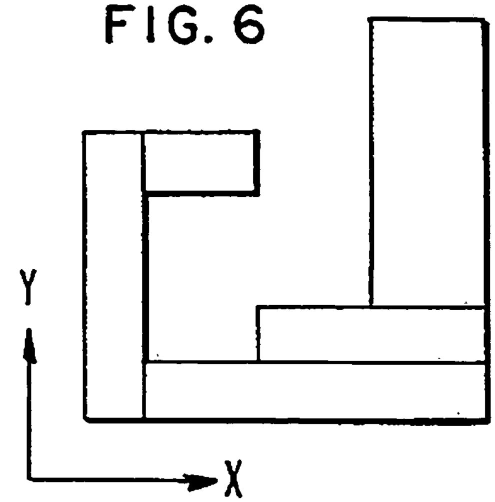


FIG. 7

MERGER
EXAMPLES

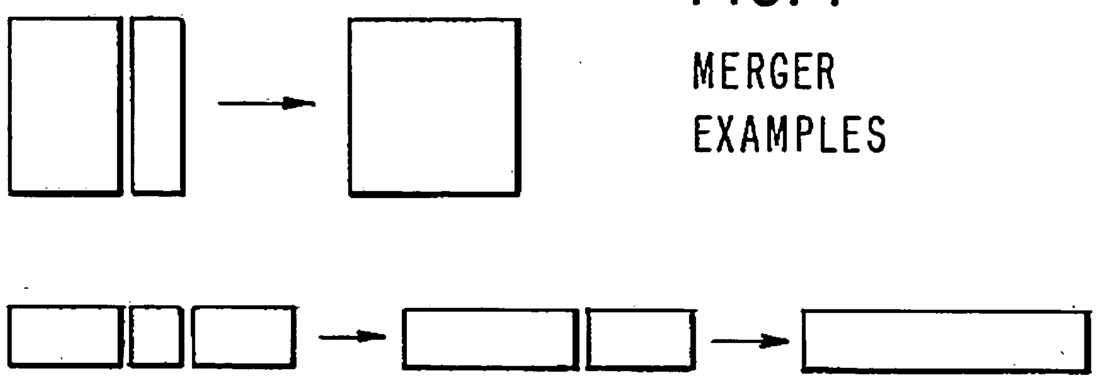


FIG. 8